JACKSON LAWRENCE
000000070612

LOUIS GABRIEL HERNANDES
000000070250

CROSS-ENTROPY OPTIMIZATION

OSCAR JIRO HARLISON
000000072786

IGNATIUS STEVEN
000000070642

DEFINITION

- **Cross-Entropy (CE) Optimization** adalah sebuah algoritma metaheuristik yang dirancang untuk menyelesaikan masalah optimasi yang dapat diestimasi dengan memperbaiki distribusi probabilitas secara iteratif pada ruang solusi
- Awalnya dikembangkan untuk simulasi dan menghitung probabilitas kejadian langka oleh Rubinstein pada tahun 1997, CE kemudian diadaptasi untuk menyelesaikan masalah optimasi yang kompleks, seperti optimasi kombinatorial, optimasi kontinu, dan machine learning

CE & MONTE CARLO

- CE sendiri merupakan perhitungan **loss antara dua distribusi probabilitas**, di mana semakin tinggi nilai CE, maka estimasi semakin acak (higher entropy)
- Dasar dari CE adalah metode **Monte Carlo**, yang digunakan untuk mencari solusi estimasi untuk masalah deterministik
- Pada dasarnya, metode Monte Carlo, yang namanya didasarkan dari kota perjudian di Monaco dengan nama yang sama, menggunakan **sampling berkali-berkali secara acak** untuk mendapatkan **hasil numerik**

WHY USE CE?

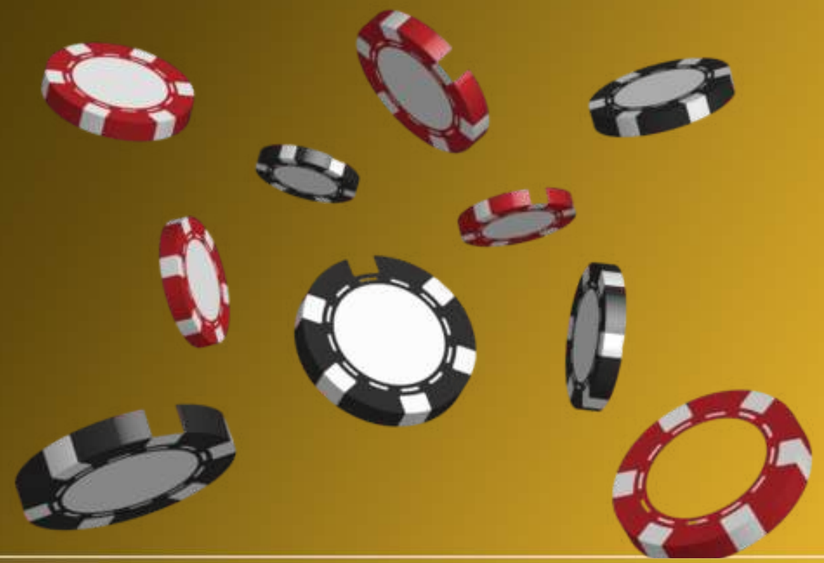
- **Simpel dan adaptif** untuk mengestimasi parameter optimal
- Dapat diaplikasikan untuk masalah optimasi **deterministik dan probabilistik**
- Cocok untuk optimasi **berbasis simulasi dan kontinu** yang kompleks
- Memiliki properti **konvergensi asimtotik** sehingga konsisten berkonvergensi dalam iterasi finit

HOWEVER...

- **Kompleks diterapkan di praktik nyata** dan memerlukan keahlian ekstensif walaupun bersifat adaptif
- **Sangat bergantung terhadap estimasi distribusi probabilitas** secara tepat
- **Memerlukan komputasi intensif** karena memerlukan volume sampel yang banyak, terutama untuk sistem kompleks

APPLICATIONS

- Buffer allocation
- Vehicle routing
- Project management
- Scheduling problems
- Queueing models of telecommunication systems
- DNA sequence alignment



HOW IT WORKS

- Awalnya, algoritma dimulai dengan suatu distribusi probabilitas pada ruang solusi
- Solusi kandidat atau **sampel acak kemudian dihasilkan dari solusi saat ini** berdasarkan parameter yang ada
- Fitness setiap sampel diuji dan yang **terbaik dipilih untuk memperbarui parameter** distribusi probabilitas
- **Sampling Monte Carlo ini diterapkan secara iteratif**, diulang terus-menerus untuk mendekati solusi optimal
- Dengan memperbarui distribusi probabilitas berdasarkan solusi terbaik dari iterasi sebelumnya, CE secara efektif mempersempit ruang pencarian, meningkatkan peluang menemukan solusi global optimal

OVERALL CONCEPT

KEY COMPONENTS & STOPPING CRITERIA

Probability Distribution

01

Mewakili kemungkinan menghasilkan solusi tertentu.

Elite Samples

02

Solusi dengan performa terbaik yang digunakan untuk memperbarui distribusi pada setiap iterasi.

Cross-Entropy

03

Ukuran perbedaan atau loss antara distribusi probabilitas saat ini dan distribusi target, yang diminimalkan untuk memandu pencarian solusi.

Sampling

04

Proses menghasilkan solusi kandidat secara acak dari distribusi probabilitas saat ini.

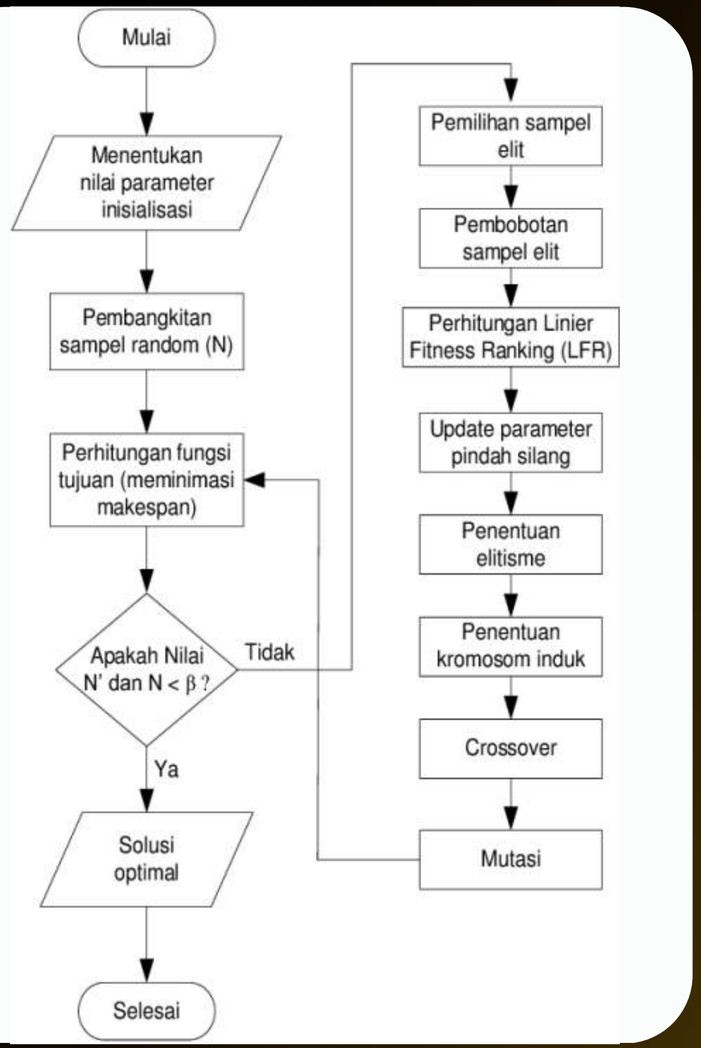
Stopping Criteria

Rata-rata perubahan nilai fitness lebih kecil dari toleransi (dianggap telah berkonvergensi.

Sudah melebihi batas iterasi maksimum.

Fungsi objektif mencapai nilai yang memuaskan.

PSEUDOCODE



Algorithm 1: Cross Entropy Optimization Algorithm

Data: Initial parameter estimate $\hat{\Theta}^0$, quantile parameter $\rho \in (0,1)$, stopping threshold ϵ, maximum iterations max_iter

Result: Optimal parameter estimate $\hat{\Theta}^*$

Initialize iteration counter $t = 1$;

while *termination criterion is not satisfied* ***and*** $t \leq max\_iter$ **do**

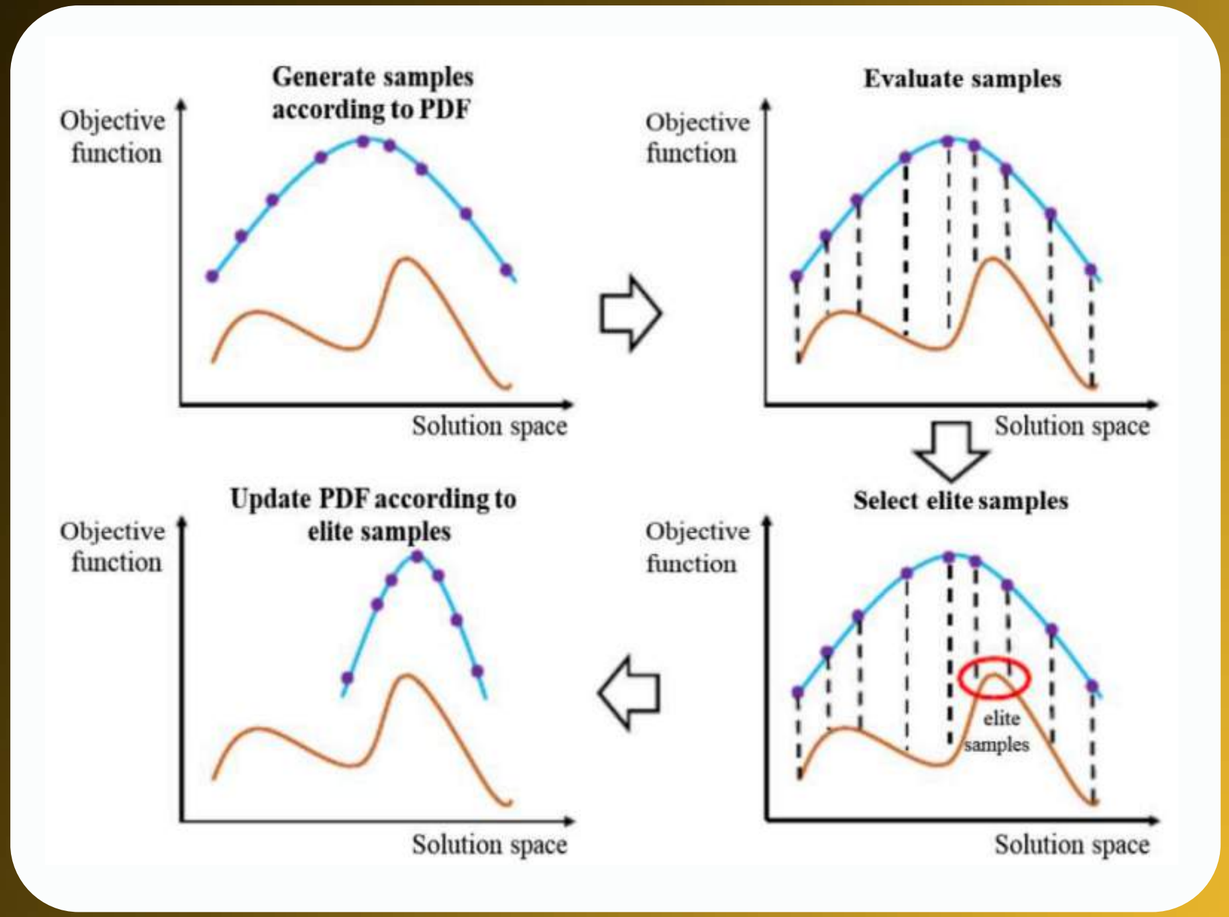
- Generate a sample $(X_1, X_2, \ldots, X_m)$ from the probability distribution $p(\cdot; \hat{\Theta}^{t-1})$;
- Compute performances $S(X_i)$ for each i;
- Sort performances $S(X_1) \leq S(X_2) \leq \ldots \leq S(X_m)$;
- Set $y^{\hat{t}} = S(\lfloor (1-\rho)N \rfloor)$;
- Solve for $\hat{\Theta}^t$ by maximizing the log-likelihood over the samples where $S(X_i) \geq y^{\hat{t}}$;
- **if** $\|\hat{\Theta}^t - \hat{\Theta}^{t-1}\| < \epsilon$ ***or*** *other termination condition is met* **then**
  - break;
- **else**
  - Increment iteration counter $t = t + 1$;
- **end**

end

Return $\hat{\Theta}^*$;

STEPS

- Inisialisasi Distribusi : Dimulai dengan mendefinisikan distribusi awal untuk variabel-variabel yang akan dioptimasi.
- Sampling : Sampel diambil dari distribusi tersebut, menghasilkan solusi-solusi potensial.
- Evaluasi : Solusi-solusi ini kemudian dievaluasi berdasarkan kriteria objektif, seperti nilai fungsi loss.
- Pembaruan Distribusi : Solusi terbaik darisampel yang diambil digunakan untuk memperbarui parameter distribusi, sehingga distribusi baru lebih condong ke arah solusi terbaik tersebut.
- Iterasi : Langkah ini diulang sampai distribusi konvergen ke solusi optimal atau mencapai kriteria penghentian tertentu.



STOPPING CRITERIA

- Convergence of Loss: Stop training when the change in loss is below a threshold (e.g., ΔLoss<0.001\Delta \text{Loss} < 0.001ΔLoss<0.001) to ensure further training won't significantly improve the model.
- Validation Loss: Use a validation dataset and stop when validation loss increases while training loss decreases (Early Stopping) to prevent overfitting.
- Maximum Epochs: Set a limit on the number of epochs to manage computational resources and ensure timely training.
- Gradient Threshold: Stop training when gradients become very small, indicating convergence.

TIME COMPLEXITY

$$O(m * N)$$

SPACE COMPLEXITY

$$O(m * N)$$

Informations:

- m : Number of instances / samples
- N : Total number of distinct categories

SHANNON ENTROPY

$$H(X) = -\sum_{i=1}^{n} p(x_i) \times \log_2 (p(x_i))$$

Informations :

- $p(x_i)$ = Probability of outcome x_i

CROSS ENTROPY

$$H(P, Q) = -\sum_{i=1}^{n} P(x_i) \times \log (Q(x_i))$$

Informations :

- H(P, Q) = Cross entropy between the probability distributions P and Q
- $P(x_i)$ = True probability distribution
- $\log(Q(x_i))$ = Predicted probability distribution

EXAMPLE CALCULATION

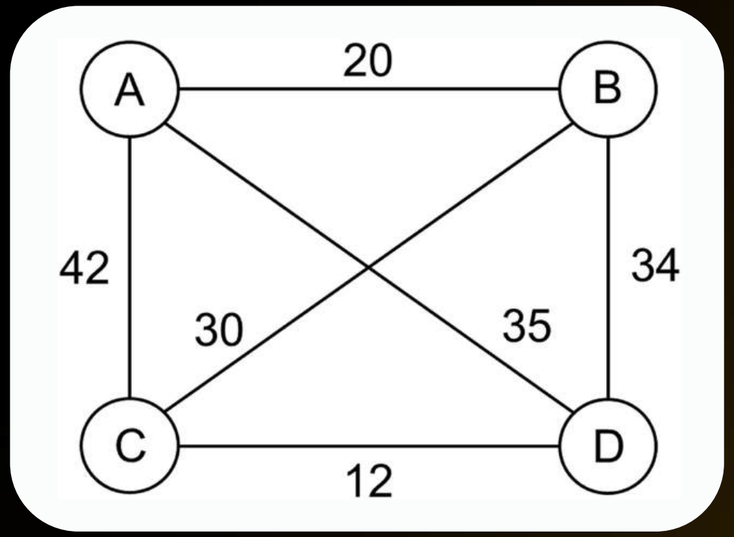
Sunny : 1%	**Sunny : 1%**	**Cloudy : 4%**	**Cloudy : 4%**
00	**01**	**100**	**101**
$1/2^2$ = 25%	**$1/2^2$ = 25%**	**$1/2^3$ = 12.5%**	**$1/2^3$ = 12.5%**

Rainy : 10%	**Rain : 10%**	**Storm : 35%**	**Storm : 35%**
1100	**1101**	**11100**	**11101**
$1/2^4$ = 6,25%	**$1/2^4$ = 6,25%**	**$1/2^5$ = 3.125%**	**$1/2^5$ = 3.125%**

H(p,q) = 1%x2 + 1%x2 + 4%x3 + 4%x3 + 10%x4 + 10%x4 + 35%x5 + 35%x5
= 4.58 bits

p = true distribution
q = predicted distrubution

TRAVELING SALESMAN PROBLEM



Probability Distribution = 1/distance

Rand(0, 1)

Cumulative Distribution Function (CDF) = P0 + P1 + ...

Iterasi 1
Step 1

A -> B -> C -> D -> A = 97
A -> D -> B -> C -> A = 141
A -> C -> D -> B -> A = 108

Step 2

Select 2 best solution
S1 = A -> B -> C -> D -> A = 97
S2 = A -> C -> D -> B -> A = 108

Step 3

Update Probability
PD1 = 1/97 = 0.01
PD2 = 1/108 = 0.009

Total = 0.01 + 0.009 = 0.019

P1 = 0.01/0.019 = 0.526
P2 = 0.009/0.019 = 0.474

CDF1 = 0.526
CDF2 = 1

Iterasi 2
Step 1

Mis rand = 0.352

Karena rand < CDF1, maka ambil S1, jika diantara CDF1 dan CDF2, maka ambil S2

Setelah memilih S1 atau S2, lakukan mutation, misal swap 2 city

Lakukan untuk setiap solution baru

Iteration 2

Step 2

A -> B -> D -> C -> A = 108
A -> D -> C -> B -> A = 97
A -> C -> B -> D -> A = 141

Step 3

Select 2 best solution
S1 = A -> D -> C -> B -> A = 97
S2 = A -> B -> D -> C -> A = 108

Step 4

Update Probability
PD1 = 1/97 = 0.01
PD2 = 1/108 = 0.009

P1 = 0.01/0.019 = 0.526
P2 = 0.009/0.019 = 0.474

CDF1 = 0.526
CDF2 = 1

Continue to next iteration

Iterasi akan berlanjut sampai mencapai stopping criteria

OPTIMIZING A FUNCTION

Misalkan sebuah fungsi $f(x) = x^2$

iteration 1	$f(x) = x^2$
$x1 = 8.8$	$x1 = 8.8^2 = 77.44$
...	...
$x20 = -9.9$	$x20 = -9.9^2 = 98.01$

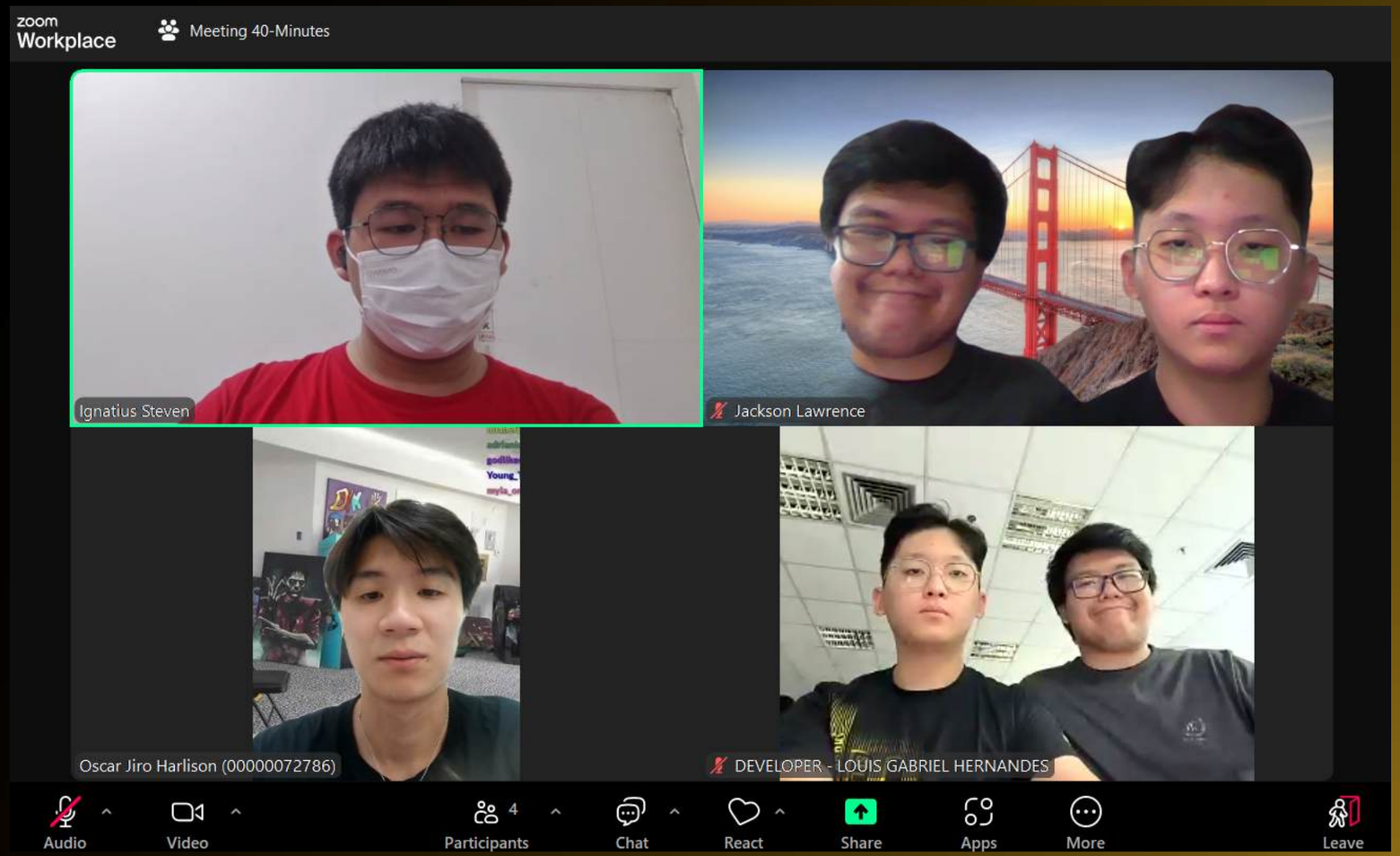
Elitism 10% = x1 & x20

$$\mu = \frac{1}{2} \times (8.8 + (-9.9)) = -0.55$$

$$\sigma^2 = \frac{1}{2} \times (8.8 + 0.55)^2 + (-9.9 + 0.55)^2 = 87.42$$

$$\sigma = \sqrt{87.42} = 9.35$$

$$N(-0.55, 9.35^2)$$



Links:

- https://www.ibm.com/topics/monte-carlo-simulation

Paper:

- https://web.mit.edu/6.454/www/www_fall_2003/gew/CEtutorial.pdf
- https://people.smp.uq.edu.au/DirkKroese/ps/aortut.pdf
- Buku Metaheruistiks dari Kevin Sorensen
- Beyonce

Video:

- https://youtu.be/ErfnhcEV1O8?si=y4x1fX9hB5mml113